Trabajo práctico N° 2:
Algoritmo de ensambles

Año 2021 - Aprendizaje Automático
Explotación de Datos y Descubrimiento de Conocimiento
Facultad de Ciencias Exactas y Facultad de Ingeniería

Grupo n° 13 - Año 2021
Bertolini, Lucas; Galíndez, Raúl

Reconocimiento de emociones a partir del habla

Resumen

En el trabajo práctico número dos trató de reconocer emociones en audios usando técnicas de ensamble de modelos de aprendizaje automático supervisados. Se partió del dataset Ryerson Audiovisual Database of Emotional Speech and Song (RAVDESS)[1] .

Armamos un pipeline para el proceso de aprendizaje automático. Dividimos en preprocesamiento para extraer las features, en exploración para entender el dataset, búsqueda de hiperparámetros para los ensambles y la evaluación de los modelos tanto con data propia del RAVDESS como con audios propios.

Datos

En el notebook “01.preprocesamiento.ipynb”, encapsulamos la lógica de la obtención del dataset para trabajar. A partir del enlace propuesto, descargamos y descomprimimos los archivos: Audio_Song_Actors_01-24.zip y Audio_Speech_Actors_01-24.zip.

Se cuenta con un dataset de 2.452 audios, correspondientes a 24 actores (12 hombres y 12 mujeres) expresando 8 emociones cada uno, tanto hablado como cantado (1440 y 1012 registros respectivamente), diciendo dos frases (01 = "Kids are talking by the door", y 02 = "Dogs are sitting by the door"), con intensidad emocional normal y fuerte, y con dos repeticiones.

Las emociones expresadas son:

- 01 = neutral
- 02 = calm
- 03 = happy
- 04 = sad
- 05 = angry
- 06 = fearful
- 07 = disgust
- 08 = surprised

Durante el análisis exploratorio[2] detectamos:

- El actor número 18 no tiene audios cantados.
- El dataset se encuentra desbalanceado, ya que las emociones ‘neutral’, ‘disgust’ y ‘surprised’ se encuentran en menor proporción. Lo anterior será importante al momento de elegir la métrica.
- No hay audios cantados para las emociones ‘disgust’ y ‘surprised’.

Para la extracción de las features de los audios se usó la librería opensmile-python usando las variables del “Geneva Minimalistic Acoustic Parameter Set”[3] en su versión 2 (eGeMAPSv02) la cual extrae parámetros de los audios relacionados con: la frecuencia, la energía/amplitud, así como parámetros espectrales (balance, dinámica, forma), en total 88 variables. Estas variables son medidas resumen de los atributos descritos anteriormente, medidos sobre los audios, entre estos se encuentran: medias aritméticas, coeficientes de variación, rangos, percentiles, ratios, entre otros. Por lo tanto, es bastante variado el rango de valores que toman estas variables.

Metodología

Los métodos de ensamble son usados para lidiar con el dilema sesgo - varianza, combinando las predicciones de distintos algoritmos para reducir el sesgo o la varianza según sea el caso. En este trabajo práctico se partió de árboles de decisión como algoritmos base para los ensambles: bagging, Random Forest (RF) y AdaBoost (AB).

Para medir el desempeño de los modelos se consideró inicialmente el Accuracy, ya que es la métrica más sencilla de la que se dispone, siendo idónea si las clases están balanceadas. Durante el análisis exploratorio encontramos ciertos desbalances por lo que preferimos usar F1 Score, la cual es una media armónica que tiene en cuenta falsos negativos y falsos positivos, y por lo tanto una métrica más robusta.

Usamos cross-validation para garantizar independencia entre las particiones de datos usadas para entrenar y validar el modelo y obtener una métrica más robusta, esto es, menos influenciada por la muestra tomada para hacer el split entre entrenamiento y test. En este caso probamos dos variantes: k-folds (StratifiedKFold) y Validación Cruzada dejando uno grupo fuera (LeaveOneGroupOut).

[1] Link para la descarga: https://zenodo.org/record/1188976#.YNjHfu1Khxd
[2] El notebook: “02.01.Analisis.exploratorio.ipynb” encapsula el análisis exploratorio.
[3] https://sail.usc.edu/publications/files/eyben-preprinttaffc-2015.pdf

Armamos un pipeline separando los distintos pasos: Preprocesamiento, Análisis Exploratorio de Datos, Búsqueda de hiperparámetros, Test de modelos y métricas, Test de audios propios. Cada paso está documentado en un notebook que encapsula esa parte del trabajo.

Resultados

Se eligió hacer bagging sin hacer búsqueda de hiper-parámetros, para tener un baseline no muy complejo, esta técnica hace resampleo con reemplazo buscando minimizar la varianza de estimadores con bajo sesgo y alta varianza.

El algoritmo base fue árboles de decisión, sin hiperparámetros, se usaron 1000 estimadores.

Para validación cruzada usamos dos métodos:

I. 12-fold cross validation (Utilizando stratified cross validation)
II. Leave-2-speakers out. Es decir, 12 folds conteniendo cada uno 2 actores distintos (Utilizar la función de sklearn LeaveOneGroupOut). Las divisiones se hicieron tomando pares consecutivos de actores, lo cual nos aseguraba que quedará en cada iteración un hombre y una mujer.

En este trabajo tomamos mayor dimensión del costo computacional en la medida que complejizamos el proceso: mayor cantidad de instancias, métodos de ensamble que se construyen a partir de otros algoritmos y mayor número de variables. Además, veníamos trabajando con problemas binarios y este fue multiclase, por lo cual la cantidad de cálculos para las métricas de desempeño aumenta (Partimos de una matriz n-aria). Véase el Gráfico 1: Comparativo de métricas para la validación cruzada en el Anexo.

Es evidente que con el segundo método tenemos mayor sesgo y mayor varianza.

Bootstrapping vs Boosting

Para complejizar el análisis se experimentó con dos modelos diferentes: Random Forests y AdaBoost. Este primer método, tiene una mejora con respecto a bagging la cual, aparte de hacer resampleo de instancias también lo hace con los disponibles. Lo anterior nos ayuda a evitar que ciertos atributos monopolicen los criterios de decisión y terminemos eligiendo entre muchos árboles muy parecidos.

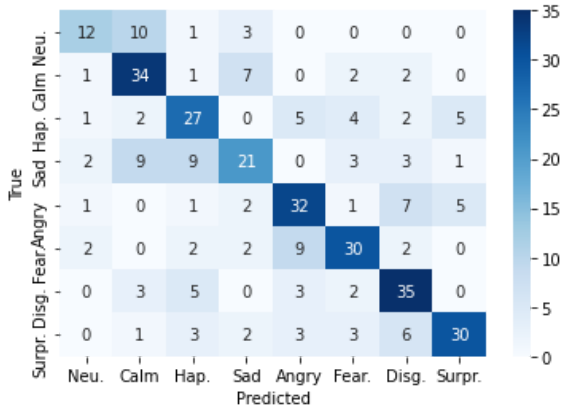
Definimos espacios de búsqueda amplios: para AdaBoost entre 10 y 3000 estimadores, y en Random Forest entre 50 y 5000. Detectamos en este escenario un problema que no habíamos enfrentado en el trabajo práctico anterior: teníamos una salida multiclase (8 emociones a predecir) y mayor cantidad de features. Cada iteración de la búsqueda de hiper-parámetros con k-fold resultaba más lenta. Optamos por 1000 iteraciones del método de sklearn: ‘RandomizedSearchCV’.

Bagging	**AdaBoost**	**Random Forest**
{‘n_estimators’:1000}	{'n_estimators': 1640, 'learning_rate': 0.707}	{'n_estimators': 3050, 'bootstrap': True}
Accuracy training : 1 Accuracy test: 0.682 Test F1: 0.67961	Accuracy training: 1 Accuracy test: 0.501 F1: 0.500927	Accuracy training: 1 Accuracy test: 0.690 F1: 0.68917

Elegimos el Random Forest dado que tuvo el mejor desempeño.

Speech

Accuracy test: 0.618
Test F1: 0.42649



Song

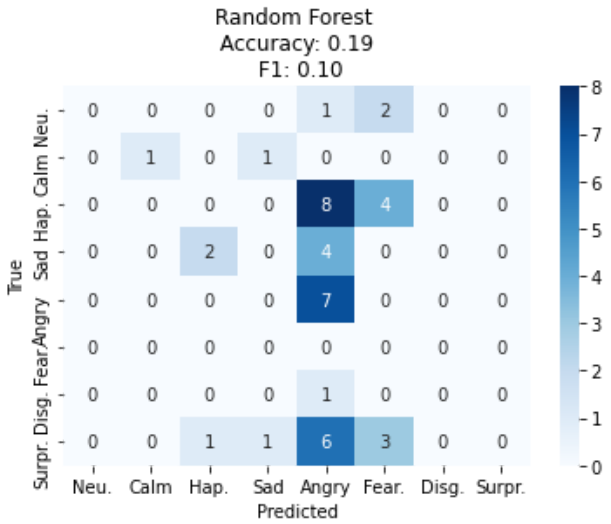
Accuracy test: 0.791
Test F1: 0.61092

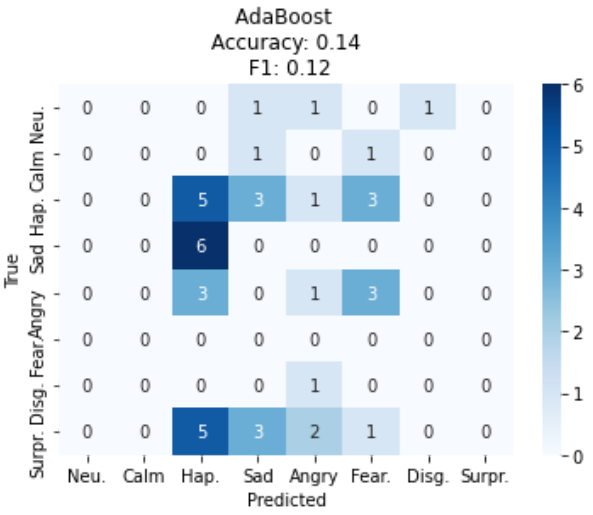
En el caso de las canciones la capacidad de predecir es mucho más alta que para los discursos. La confusión entre pares de emociones es baja. Las emociones “neutral” y “angry” fueron clasificadas con alta eficacia.

Cuando hablamos de discurso, el desempeño baja considerablemente. El score, 0.61, es un 45% mayor que 0.42. También podemos apreciar cómo la emoción “neutral” tiende a ser mal clasificada como “calm”.

Desempeño del modelo con grabaciones propios

Si bien se usan diversas técnicas como el kfold cross-validation para tener una medida robusta del desempeño del modelo, y luego se prueba con un conjunto no visto por el algoritmo en el entrenamiento, es posible, por diversa razones, que al poner a prueba el modelo con datos de un entorno real, menos controlado, como el dispuesto en el dataset de RAVDESS surjan diferencias importantes en las métricas. Se probó con el mejor modelo obtenido según lo obtenido con data del dataset proporcionado, a saber, un Random Forest, además del modelo con AdaBoost. A continuación los resultados:





Ambos modelos tienen un desempeño muy por debajo de lo que fue con la data de RAVDESS. Además, aunque inicialmente Random Forest tuvo mejores métricas que AdaBoost, para los audios propios se da lo contrario.

Las predicciones de Random Forest se concentraron, erróneamente, en la emoción: “enojo”. Dicha emoción fue la mejor predicha, es decir, en cuanto a completitud aunque también la más confundida, ya que 74% de las predicciones fueron incorrectas. Este error de clasificación se concentró en felicidad y sorpresa.

Por su parte el AdaBoost tuvo mejor desempeño clasificando audios felices y el error mayor fue en la tristeza, contradictoriamente, no logró detectar los audios tristes.

La combinación de un dataset de entrenamiento con solo 12 personas (más allá de que expresan distintos sentimientos de distintas formas, intensidades y otras variantes), aunado al hecho de que las grabaciones propias fueron hechas por personas con nulas habilidades histriónicas hacen que el modelo se haya desempeñado pobremente. Dado que el desempeño de estos modelos no superó 0.12, mientras que para el dataset de validación de los datos originales fue de 0.50 y 0.69 (AB y RF respectivamente) se puede afirmar que el modelo tiene escaso poder de generalización frente a distintas frases, formas de pronunciación y otras variantes.

Siguiendo estas conclusiones pensamos en otra línea de investigación que podríamos encarar: stacking. Si lográramos modelos que funcionan mejor con emociones específicas podríamos armar un meta-modelo con stacking y mejorar la capacidad predictiva. Lo anterior queda como una hipótesis a validar.

Consideraciones finales

Durante el trabajo práctico experimentamos con tres métodos de ensamble con una forma de resolver el problema diferente. Lo que es interesante destacar es que en los tres modelos de ensamble utilizados tuvimos sobreajuste, teniendo siempre un accuracy para el entrenamiento de 1, y en test fue mucho menor: entre 0.4 y 0.68.

Viendo este escenario nos preguntamos, ¿Cuál es el equilibrio entre el desempeño obtenido y el tiempo invertido? Aún con búsquedas de hiper-parámetros de muchas horas, las mejoras en las métricas obtenidas eran bajas, aproximadamente una décima o una centésima.

Pudimos ver la incapacidad de generalizar de nuestro modelo en dos oportunidades: inicialmente al usar el método LeaveOneGroupOut, donde vimos cómo al dejar dos participantes afuera del entrenamiento aumentaba considerablemente el sesgo y varianza, y luego al utilizar audios grabados por integrantes del grupo donde obtuvimos métricas de desempeño ligeramente mejores que el azar.

La hipótesis que surge, para trabajos posteriores, es si necesitamos buscar nuevos caminos: obtener más datos que represente más ampliamente la variedad de personas y la forma en que expresan sus emociones, formas alternativas de extraer características significativas de los audios o probar nuevos algoritmos con mayores capacidades explicativas.

Hicimos unas pruebas con los atributos del Compare_2016 (Computational Paralinguistics ChallengE)[4], otra forma que ofrece opensmile para extraer features de los audios. Usamos los mismos dos modelos a los que llegamos en la sección “Bootstrapping vs Boosting”, obteniendo métricas peores que las que obtuvimos con Random Forest y AdaBoost (Un F1-score 0.10 y 0.12 respectivamente). Nos encontramos con algunas dificultades para hacerlo funcionar, con la librería opensmile y los cambios recientes que tuvo; y también con el poder de cómputo: el hardware de las computadoras personales no fue suficiente. No se pudo entender a qué correspondía esa disminución en las métricas, probablemente se deba a la cantidad de estimadores, la cual pudiera ser suficiente para un conjunto de datos con 90 columnas, pero no suficiente para un conjunto de datos con más de 6000 atributos.

Algo similar nos pasó con la implementación de redes neuronales, lo cual es desalentador para este trabajo práctico, pero sí deja el camino marcado para seguir probando y experimentando. Siendo la primera vez que nos enfrentamos a audios, con métodos de ensambles y problemas de clasificación multiclase, vemos un balance muy positivo. Cumpliéndose el objetivo número uno que era poder poner en práctica, en un problema casi-real, lo dado a lo largo de la cursada de la materia; y también habiéndose abierto muchas líneas para continuar explorando.

[4] Notebook: “06.Metricas_compare.ipynb”.

Anexo

Gráfico 1: Comparativo de métricas para la validación cruzada